Chapter 3

Neural Network Modelling

Regression analysis is familiar to scientists as a tool to fit experimental data empirically . The linear relationship is chosen before the best–fit coefficients are derived. The general form of the equation developed using linear regression is a sum of the inputs x_i multiplied by a corresponding coefficient or weight w_i and added with a constant (θ). The developed linear equation may contain non–linear terms, forming a pseudo–linear equation. In linear regression models the relationship between a input and output tends to be linear and applies across the entire span of the input space, which may not be reasonable. Neural networks is a general method of non–linear regression which avoids the difficulties occurs in linear regression technique. In this Chapter the fundamentals of neural networks and procedure followed to develop models are discussed.

3.1 Neural Networks

A neural network is a general method of regression analysis in which a very flexible non–linear function is fitted to experimental data. When compared with linear regression analysis, neural networks is a non–linear regression by introducing an another node which is hidden in between input and output as shown Fig 3.1. Similar to linear regression method the input variable x_i is multiplied by weight w_i^1, but the sum of all these products forms the argument of a another transfer function, in this present work it is hyperbolic tangent as in equation 3.2. The final output is defined as linear function of hidden nodes and a constant, equation 3.1. Thus, the dependent variable y is defined as;

$$y = \sum_i w_i^{(2)} h_i + \theta^{(2)}, \tag{3.1}$$

where h_i defined as;

$$h_i = \tanh\left(\sum_j w_{ij}^{(1)} x_j + \theta_i^{(1)}\right) \tag{3.2}$$

where x_j are the j variables on which the output y depends, w_i are the weights (coefficients) and θ_i are the biases (equivalent to the constants in linear regression analysis). The combination of equation 3.2 with a set of weights, biases, value of i and the minimum and maximum values of the input variables defines the network completely, Fig. 3.1. The availability of a sufficiently complex and flexible function means that the analysis is not as restricted as in linear regression where the form of the equation has to be specified before the analysis. The strength of the hyperbolic tangent transfer function is determined by the weight w_j, the exact shape can be varied by altering the weights. The shape of the hyperbolic transfer function will be varied according to the availability of data in the input space. A model with one hidden unit (Fig. 3.2a) may not sufficiently flexible to capture the information from the database, however non–linearity can be increased by combining several of the hyperbolic tangents as shown in Fig. 3.2b.

The neural network can capture interactions between the inputs because the hidden units are nonlinear. The nature of these interactions is implicit in the values of the weights, but the weights may not always be easy to interpret. For example, there may exist more than just pairwise interactions, in which case the problem becomes difficult to visualise from an examination of the weights. A better method is to actually use the network to make predictions and to see how these depend on various combinations of inputs.

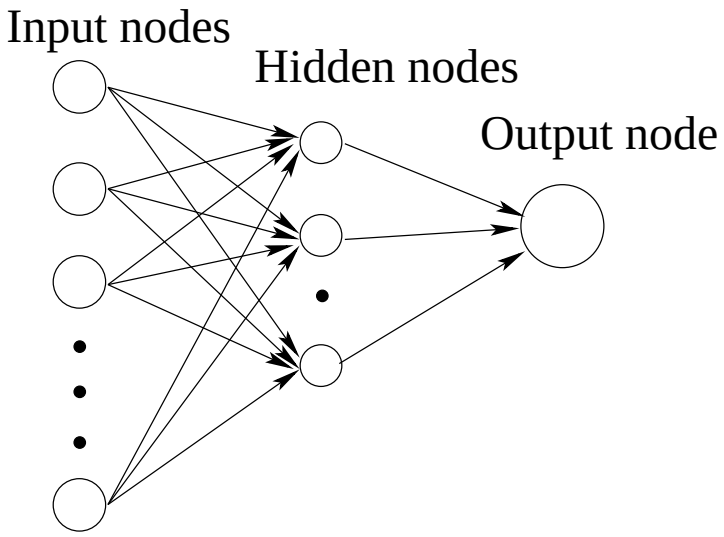

Figure 3.1: Schematic illustration of input, hidden and output layers of neural network model used in the present work.

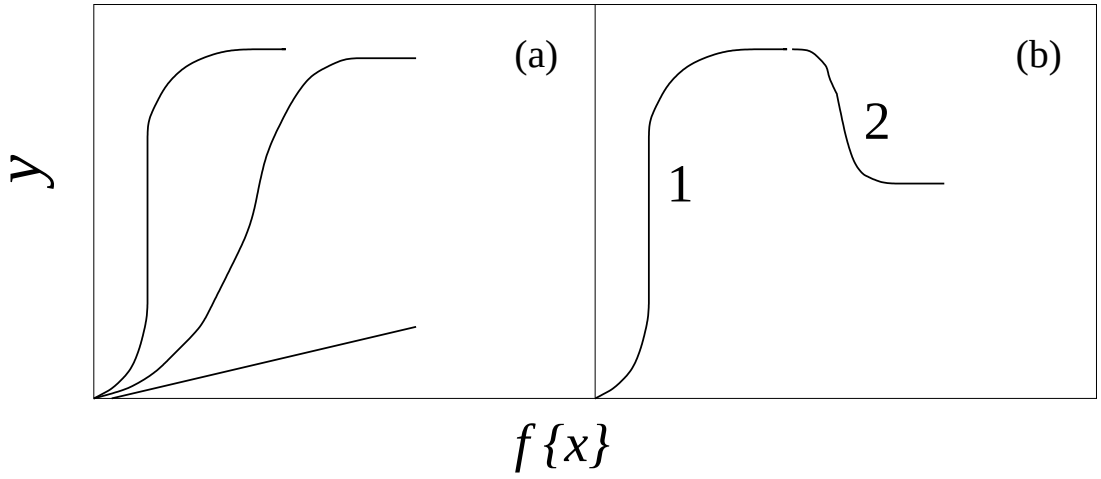

Figure 3.2: Hyperbolic tangent relation between inputs x and output y, a) single flexible hyperbolic tangent with varying weights b) combination of two tangents.

3.2 Error Estimation

The input parameters are generally assumed in the analysis to be precise and it is normal to calculate an overall error by comparing the predicted values (y_j) of the output against those measured (t_j), for example,

$$E_D \propto \sum_j (t_j - y_j)^2 \tag{3.3}$$

E_D is expected to increase if important input variables have been excluded from the analysis. Whereas E_D gives an overall perceived level of noise in the output parameter, it is, on its own, an unsatisfying description of the uncertainties of prediction.

MacKay has developed a particularly useful treatment of neural networks in a Bayesian framework [6], which allows the calculation of error bars representing the uncertainty in the fitting parameters. The method recognises that there are many functions which can be fitted or extrapolated into uncertain regions of the input space, without unduly compromising the fit in adjacent regions which are rich in accurate data. Instead of calculating a unique set of weights, a probability distribution of sets of weights is used to define the fitting uncertainty. The error bars therefore become large when data are sparse or locally noisy.

In this context, a very useful measure is the log predictive error because the penalty for making a wild prediction is reduced if that wild prediction is accompanied by appropriately large error bars [6]:

$$\text{LPE} = \sum_m \left[\frac{1}{2} \frac{(t^{(\text{m})} - y^{(\text{m})})^2}{\sigma_y^{(\text{m})^2}} + \log\left(\sqrt{(2\pi\sigma_y^{(\text{m})})}\right) \right] \tag{3.4}$$

where $\sigma^{(\text{m})}$ is the error bar calculated using Bayesian statistics [6]. A larger value of the log predictive error implies a better model, Fig 3.4b.

3.3 Overfitting

A potential difficulty with the use of powerful non–linear regression methods is the possibility of overfitting data. To avoid this, the experimental data can be divided into two sets, a *training* dataset and a *test* dataset. The Fig. 3.3 illustrates different degrees of complexity in fitting the training dataset and the test data. A linear model is simple and does not capture the real information form the data. An overcomplex model fits all the data in the training dataset, but badly generalised. The optimum model which is a generalised model captures real complexity in the database, Fig. 3.3.

The model is produced using only the training data. The test data are then used to check that the model behaves itself when presented with previously unseen data. The training error tends to decrease continuously as the model complexity increases, Fig 3.4a. It is the highest log predictive error (Fig 3.4b) which enables that model to be chosen which generalises best on unseen data [6].

The analysis uses normalised values of the variables in the range ± 0.5 as follows:

$$x_N = \frac{x - x_{min}}{x_{max} - x_{min}} - 0.5 \tag{3.5}$$

where x is the original value from the database, x_{max} and x_{min} are the respective maximum and minimum of each variable in the original data and x_N is the normalised value. This step is not essential to the running of the neural network but is a convenient way of comparing the effect of different variables on the output. Fig. 3.1 shows the general structure of the simple three layer neural network.

3.4 Model Development Procedure

The experimental data collected are stored in a particular format. These data are normalised using equation 3.5. The normalisation of experimental data is not necessary for the development of models, but it helps in comparing the relative influence of different input variables. Around 80 different neural network models are selected for training over chosen functions (Equ 3.1 and 3.2). These models will differ in number of hidden units and seed to generate random starting weights. Before 'training' of the model, the experimental database is randomised in order to divide the

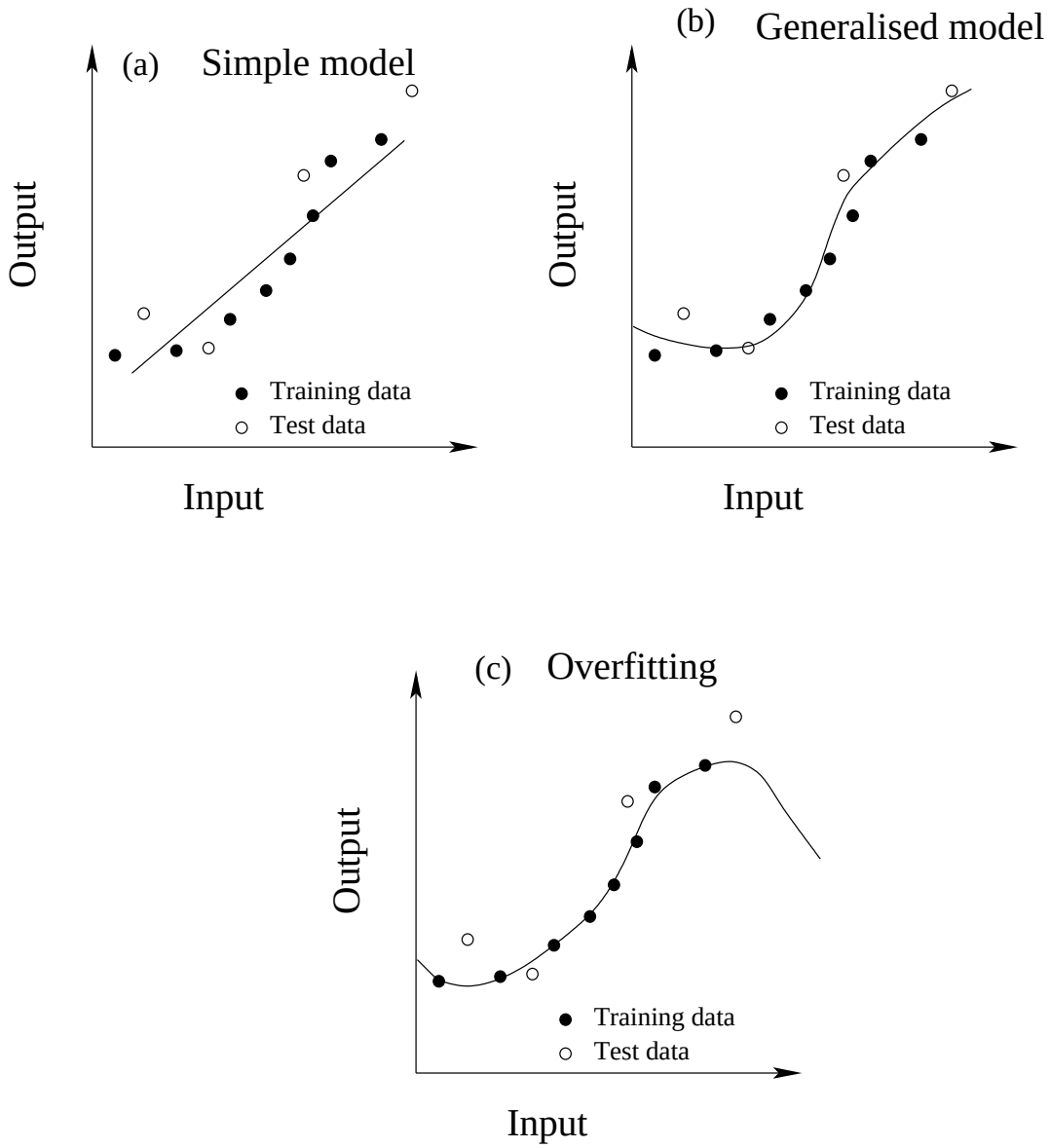

Figure 3.3: Different degrees of complexity of fitting a input and output in a model.

information into test and training datasets in a fair manner. The first half of the randomised dataset is used for training and the remaining is for testing how the trained models behave with unseen data.

For a trained model with database 'D', the overall error 'E_{D}' is the sum of squared error between the desired output (target) 't' and calculated output 'y, equation 3.6.

$$E_{\mathrm{D}} = \frac{1}{2}\sum_{m} \left(t^{(\mathrm{m})} - y^{(\mathrm{m})}\right)^2 \tag{3.6}$$

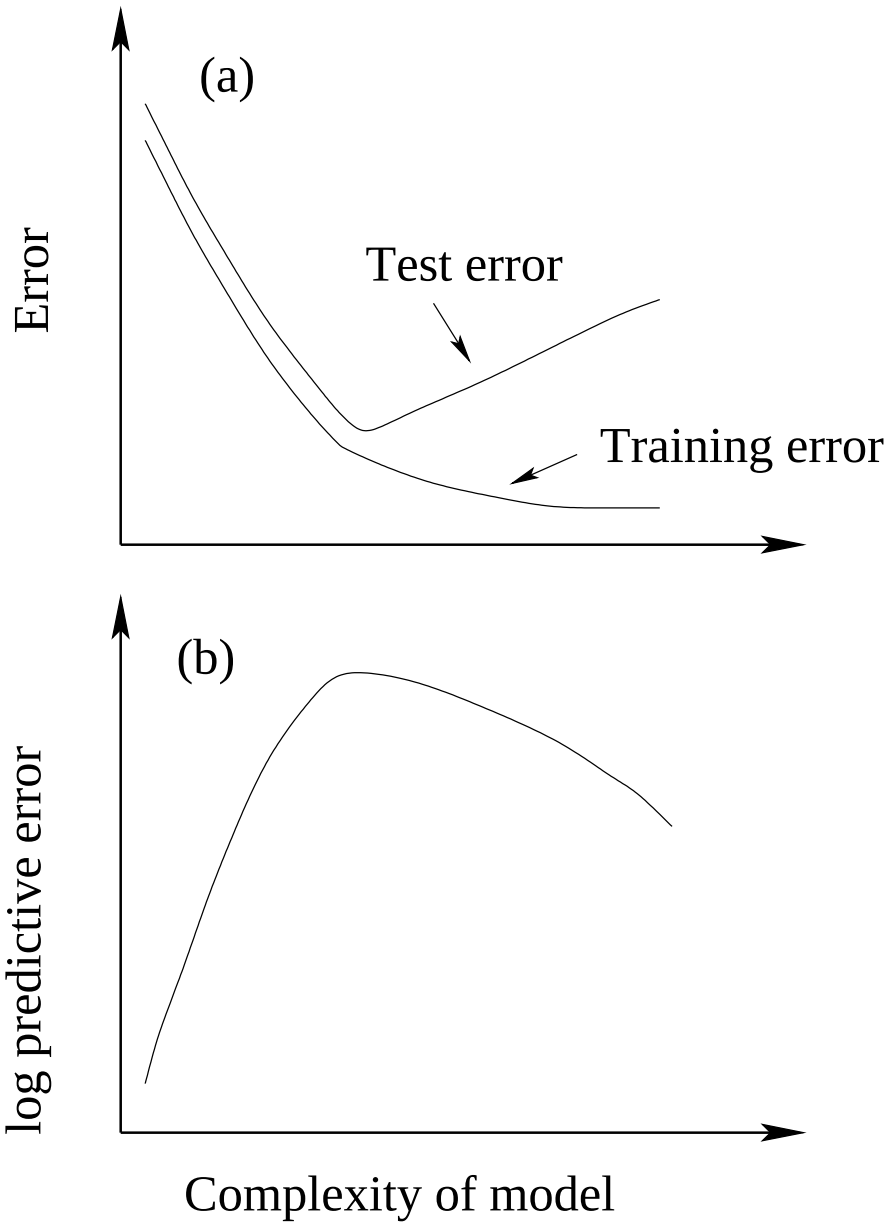

Figure 3.4: Ranking procedure of trained models with varying model complexity a) considering the variation in the test and training error b) log predictive error [6].

Then to have predictions with error bars, the trained models are ranked with decreasing magnitude of log predictive error. It is possible that a committee of models can make a more reliable predictions than an individual model [6]. Starting from the best model, the committee models are selected until the minimum validation or test error is obtained. The committee prediction is the average value of individual model predictions. During predictions using the committee model containing 'L' individual models, average output ($\bar{y}$) and the committee error

bar (σ) are calculated using following equations;

$$\bar{y} = \frac{1}{\mathrm{L}} \sum_{l} y^{(l)} \tag{3.7}$$

$$\sigma^2 = \frac{1}{L} \sum_{l} \sigma_y^{(l)^2} + \frac{1}{L} \sum_{l} (y^{(l)} - \bar{y})^2 \tag{3.8}$$

Without changing the complexity of individual models, the committee is retrained on whole database. During the retraining the weights are adjusted to better fit whole database.

The committee model predictions are the average of calculated values of each individual model in the committee. The architecture (hidden units, seed, etc.) of committee model is complex. This complexity is considered by the neural networks during the training and testing of each individual model. The committee model does not contain any information about any perceived significance of each individual input variable over the output variable like individual model, but the only way to know the effect of each input variable on output is by doing predictions for a given set of input variables and varying the single input variable over a range. Note that error bars have to be taken into consideration during the predictions.

3.5 Interpretation

The neural network can capture interactions between the inputs because the hidden units are non–linear. The nature of these interactions is implicit in values of weights, which difficult to interpret. Interpretation is best done by making predictions and examining the trends taking error bars into consideration.

These error bars, which are calculated using Bayesian inference [6] have special meaning when compared with regression analysis error calculations. As shown in Fig. 3.5, the error bar is a measure of uncertainty in fitting parameters in the noisy data region (A) or the warning message generated when it is making calculations in the region of input space where the data (with which it was trained) are sparse (B). Thus error bars calculated using Bayesian neural network represents both experimental noise and the uncertainty in prediction due lack of information in that data range. The models developed using neural networks are discussed in next chapter.

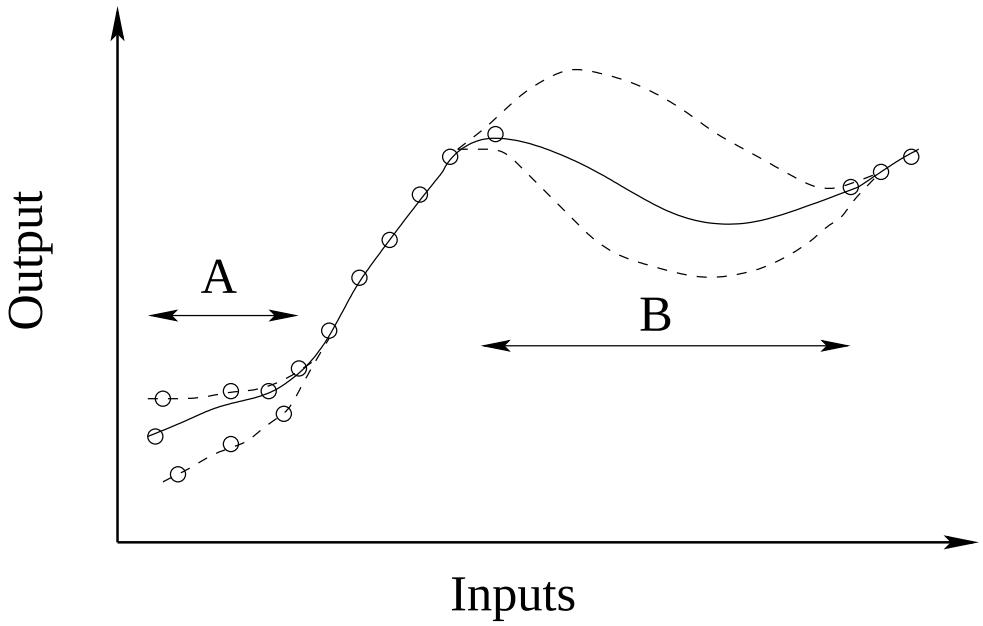

Figure 3.5: Schematic illustration of the uncertainty in defining a fitting function in regions where data are sparse (B) or where there is scatter (A). The dashed lines represent error bounds due to uncertainties in determining the weights.